Automatic discovery and ranking of synonyms for search keywords in the web

K.C. Srikantaiah*, M.S. Roopa,
N. Krishna Kumar and K.R. Venugopal

Department of Computer Science and Engineering,
University Visvesvaraya College of Engineering,
Bangalore University,
Bangalore, India
Email: srikantaiahkc@gmail.com
Email: roopams22@gmail.com
Email: krishnakumarn@live.com
Email: venugopalkr@gmail.com
*Corresponding author

L.M. Patnaik

DESE,
Indian Institute of Science,
Bangalore 560012, India
Email: patnaiklm@yahoo.com

Abstract: Search engines are an indispensable part of a web user's life. A vast majority of these web users experience difficulties caused by the keyword-based search engines such as inaccurate results for queries and irrelevant URLs even though the given keyword is present in them. Also, relevant URLs may be lost as they may have the synonym of the keyword and not the original one. This condition is known as the *polysemy* problem. To alleviate these problems, we propose an algorithm called automatic discovery and ranking of synonyms for search keywords in the web (ADRS). The proposed method generates a list of candidate synonyms for individual keywords by employing the relevance factor of the URLs associated with the synonyms. Then, ranking of these candidate synonyms is done using co-occurrence frequencies and various page count-based measures. One of the major advantages of our algorithm is that it is highly scalable which makes it applicable to online data on the dynamic, domain-independent and unstructured World Wide Web. The experimental results show that the best results are obtained using the proposed algorithm with WebJaccard.

Keywords: candidate synonym; hyperlink; inbound anchor text; ranking; search engine; similarity measures.

Reference to this paper should be made as follows: Srikantaiah, K.C., Roopa, M.S., Krishna Kumar, N., Venugopal, K.R. and Patnaik, L.M. (2014) 'Automatic discovery and ranking of synonyms for search keywords in the web', *Int. J. Web Science*, Vol. 2, No. 4, pp.218–236.

Biographical notes: K.C. Srikantaiah is a Professor in the Department of Computer Science and Engineering at SJB Institute of Technology, Bangalore, India. He obtained his BE, ME, and PhD in Computer Science and Engineering from Bangalore University, Bangalore. His research interest is in the area of data mining, web mining and semantic web.

M.S. Roopa is an Assistant Professor in the Department of Computer Science and Engineering at Impact College of Engineering and Applied Sciences, Bangalore, India. She obtained her BE in Medical Electronics from Visvesvaraya Technological University, Belgaum, India and ME in Information Technology from Bangalore University, Bangalore. Her research interest is in the area of data mining, web intelligence and big data.

N. Krishna Kumar received his BE in Information Science and Engineering and MTech in Computer Engineering from Visvesvaraya Technological University, Belgaum, India, in the years 2010 and 2012, respectively. His research interests include web mining, web technologies and computer networks.

K.R. Venugopal is currently the Principal, University Visvesvaraya College of Engineering, Bangalore University, Bangalore. He obtained his Bachelor of Engineering from University Visvesvaraya College of Engineering. He received his Masters in Computer Science and Automation from Indian Institute of Science Bangalore. He obtained his PhD in Economics from Bangalore University and PhD in Computer Science from Indian Institute of Technology, Madras. He has authored and edited 51 books. During his three decades of service at UVCE he has over 400 research papers to his credit. His research interests include computer networks, wireless sensor networks, parallel and distributed systems, digital signal processing and data mining.

L.M. Patnaik is currently Honorary Professor, Indian Institute of Science, Bangalore. He was a Vice Chancellor, Defense Institute of Advanced Technology, Pune, India. He was a Professor since 1986 with the Department of Computer Science and Automation, Indian Institute of Science, Bangalore. During the past 35 years of his service at the institute he has over 700 research publications in refereed international journals and refereed international conference proceedings. He is a fellow of all four leading science and engineering academies in India. His areas of research interest have been parallel and distributed computing, mobile computing, CAD for VLSI circuits, soft computing and computational neuroscience.

This paper is a revised and expanded version of a paper entitled 'Automatic discovery of synonyms from the web based on inbound anchor text' presented at 7th International Conference on Data Mining and Warehousing (ICDMW), Bangalore, India, 9–11 August 2013.

1 Introduction

The World Wide Web (WWW) is a collection of interconnected web pages accessed via internet that provides information and services from all over the world. The search engine is a web tool that accepts query keywords as inputs, searches the keywords in its database and provides the pages that contain these keywords as search engine result pages (SERPs). Search engines have become an indispensible part of a web user's life as they have revolutionised the internet usage by making tasks such as information retrieval and searching very easy and fast. Over the years, many schemes have been proposed to further enhance the features of the search engines and one such technique is the generation of synonyms for search keywords to improve the efficiency of the engine and accuracy of the search results.

Also, searching for the information about people is a common activity in the internet and it is a highly ambiguous task because a single name tends to be shared by many people. A traditional search engine such as *Google* and *Yahoo* would return web pages in response to the search keyword entered (the person name in this case) leaving the burden of disambiguating and collecting pages relevant to a particular person (among the namesakes) on the user. A person is generally referred by multiple name aliases on the web. Information retrieval about people from web search engines can become difficult when a person has nicknames or name aliases. Synonyms are important for solving these problems and also various other difficulties experienced by users in the field of natural language processing (NLP) such as text summarisation, question answering, text generation, search query expansion, etc. Hence, it can be seen that synonyms of search queries are quite essential.

1.1 Motivation

Search engines are undoubtedly one of the best keyword-based tools for information retrieval. They generate SERPs that contain numerous links to the URLs associated with the keyword. However, one major drawback of such a mechanism is that it depends heavily on the keyword to search for the relevant URLs. In other words:

1 irrelevant URLs that contain the same keywords may inadvertently be listed among the SERPs

2 a page that contains information relevant to the query, but does not contain the keyword, will not be listed by the search engine (*polysemy* problem) and thus in both the cases, the user may not find the appropriate page.

The *polysemy* problem may be solved if the user enters both the keyword and the synonym, but the user may not know the synonyms for all the keywords. Hence, the proposed method can be deployed to extract synonyms from the web automatically.

1.2 Contribution

In this paper, we propose a dynamic, online domain-independent algorithm called automatic discovery and ranking of synonyms for search keywords in the web (ADRS) that provides a ranked list of synonyms by first generating candidate synonyms. We have generated the candidates by comparing the URLs obtained in the SERPs by querying both the original keyword and its subsequent results. The key strategy of our approach is to extract the inbound anchor text as a candidate synonym when a potential match occurs on comparison of these corresponding initial and subsequent URLs. Finally, different similarity measures based on co-occurrence frequencies and page counts have been employed for ranking the candidate synonyms and furthermore, we draw comparisons among these ranking schemes to find out the best method that gives the most relevant synonyms.

1.3 Organisation

The remainder of this paper is organised as follows. In the next section, the related works and existing techniques are discussed briefly. In Section 3, we explain the background of this paper and the proposed system architecture. Section 4 contains problem definition and the mathematical model where we explain ADRS in detail. Section 5 provides the ranking of the candidate synonyms with respect to various measures. Finally, conclusions are presented in Section 6.

2 Related work

Extensive research in the field of web information retrieval has been achieved. However, not much effort has been put into finding people from the web. Harada et al. (2004) have proposed a method named entity extraction and association search (NEXAS) for finding authoritative people from the web by associating a web page through identification of its real world entities and determines the most relevant entities considering the top-ranked web pages from SERPs. Four simple scoring functions: *document frequency* (*df*), *server frequency* (*sf*), *document frequency and inverse document frequency* (*dfidf*) and *server frequency and inverse document frequency* (*sfidf*) are used to calculate the relevance score to rank each entity. This approach is useful to see social networks reflected on the web without explicit mentioning of relationships. However, this technique does not focus on finding other entities than people and does not take into account the co-occurrence relationships among the extracted personal names.

Kalashnikov et al. (2008) proposed a system web entities search technologies (WEST) that implements two algorithms: a graph-based disambiguation algorithm for disambiguating among people who have the same name and a graph-based cluster algorithm to improve web people search by presenting to the user a set of clusters of web pages, one cluster per distinct person. Each cluster contains all the web pages related to that person and allows the user to select the cluster of that person of interest. Using this method, the web pages of the people who are not popular and which were overshadowed in the traditional search engine will be made visible to the user. However, this technique does not consider external data sources such as ontologies, encyclopedia, etc. for disambiguating among people and also these algorithms are domain-specific.

Since in WEST, the number of people in the shared data set is not known in prior, Lefever et al. (2009) describe a fuzzy ant-based algorithm that does not require prior specification of the number of clusters, which makes it very well-suited for the web people search task. This technique is shown to be more robust than the agglomerative hierarchical clustering (Agnes) and k-means clustering algorithm. Balog et al. (2008) proposed a person clustering hypothesis which states that similar documents tend to represent the same person. It is used for disambiguating a person name in a web searching and employs two clustering techniques: single pass clustering (SPC) and probabilistic latent semantic analysis (PLSA).

In the web people search, in order to improve the search quality and provide better SERPs for the user it is important to classify web queries to know whether the query is a personal name or not. Shen et al. (2008) designed an approach to predict whether a web query is a personal name, without referring to any other context information. Personal name classification in web queries works under two stages: offline stage and online stage. During the offline stage, probabilistic name-term dictionary is constructed for a given list of candidate name terms and during the online stage, the probability of the query being a personal name is computed, based on the constructed probabilistic dictionaries and some grammars. This technique outperforms the traditional approach in terms of F-score.

Jiang et al. (2009) proposed the graph-based framework for disambiguating people appearances in web search (GRAPE) to resolve ambiguity in the people search. This technique first extracts people tag information such as name, organisation, e-mail address, etc., from the search results and a graph is modelled on the extracted tags. Finally, a clustering algorithm is performed on the graph to cluster all the extracted tags for each people entity. To solve a similar problem, Smirnova et al. (2010) used the link structure of the web pages (web graph) to disambiguate personal names using clustering algorithms.

When a person has nicknames or named aliases, the information retrieval about that person from web search engines can become difficult. To solve this problem, Bollegala et al. (2008) devised a lexical-pattern-based approach to automatically discover personal name aliases from the web. In this technique, the candidate aliases of a given name are extracted from snippets present in the SERPs of a name. These candidate aliases are ranked using three ranking approaches: lexical pattern frequency, co-occurrences in anchor-texts and page counts-based association measures. This technique was shown to significantly improve the recall rate in a relation detection task and also outperforms the traditional alias extraction methods. To identify second or higher order associations between a name and candidates aliases, Bollegala et al. (2011b) proposed an approach, a co-occurrence graph-based approach. In this method, first an undirected co-occurrence graph is constructed. However, this approach does not extract aliases for other entity types such as products, organisations and locations.

Shen and Boongoen (2012) presented a fuzzy set-based qualitative approach model called absolute order-of-magnitude (AOM) for detecting aliases that incorporates multiple link properties such as cardinality and uniqueness to evaluate the similarity between information objects for the given entities and their associations. This method outperforms several methods over datasets available in the crime/terrorism-related publication and e-mail domains. However, this technique has not been evaluated with more relevant datasets.

Finding synonyms from the web is a challenging task as they can be associated with general terms unlike aliases that apply only to people. Kawai et al. (2012) proposed a synonym extraction method in specification document by considering the co-occurrence words of component words. Simanovsky and Ulanov (2011) proposed a pattern extraction algorithm to extract the text fragments between pairs of synonyms by exploiting on large scale repositories, namely Wikipedia. In order to apply pattern extraction to Wikipedia, this technique builds a set called ‘synonymous phrases’ or mark-up on Wikipedia articles by considering the redirect pages titles, anchor text of back-links and disambiguation pages titles. Next patterns are extracted, measured and ranked according to their confidence levels. Finally, these patterns can be used for extracting synonyms from the free text.

Niemi et al. (2012) proposed a bilingual resource method for finding new synonym candidates and these are added to the existing synonym sets (synsets) of a wordnet. This technique automatically extracts groups of synonyms yielding a high number of synonyms with significant accuracy. However, the accuracy of synonym candidates, which have several possible target synsets, needs to be improved.

Van der Plas and Tiedemann (2006) describe a distribution similarity measure using automatic word alignment for finding synonyms using two different resources, a large monolingual corpus and a multilingual parallel corpus in 11 languages. Monolingual syntax-based approaches use grammatical relations to determine the context of a target word and assume that the words that share grammatical relational contexts are semantically related. However, this approach has proven to be quite successful for finding semantically related words. Multilingual alignment-based approach translates a target word into other languages found in parallel corpora and defined that as the context of target word and assumed that words that share translational contexts are semantically related. But translation will yield less semantically related words because translations do not extend to hypernyms, or hyponyms, or antonyms. Multilingual alignment-based approach extracts synonyms with much greater precision and recall when compared to monolingual syntax-based method.

Takeuchi (2008) proposed a graph-based co-clustering approach called bipartite graph algorithm to extract verb and noun synonyms by considering co-occurrences of verbs and their arguments from large scale texts. This method achieves a higher accuracy than those of a vector-based single clustering approach. Ageishi and Miura (2010) presented statistical machine translation (SMT) technique to automatically extract domain specific synonyms by considering pairs of sentences from two corpuses in Japanese and English languages and relies on word alignment to estimate translation probabilities.

Several similarity measures have been devised to find the synonyms and aliases. Li et al. (2003) proposed a new similarity measure which is a combination of shortest path length and depth of subsume nonlinearity to measure the semantic similarity between words. First, this method preprocesses the first hand information from different sources. Next, words are compared within a closed interval of *completely similar* and *nothing similar*. Finally, by using this similarity measure, the appropriate semantic words are generated.

Iosif and Potamianos (2010) proposed an unsupervised context-based similarity computation algorithm to compute semantic similarity between words using web documents. This technique first downloads the top-ranked documents returned by a web query and then computes the frequency of occurrence of contextual features. This algorithm does not refer to any external knowledge resources and can be generalised and applied to different languages. This method significantly outperforms the page-count-based metrics in terms of correlation scores. However, this algorithm does not take into consideration several issues such as document selection, feature selection and feature fusion.

Li et al. (2006) presented a text similarity algorithm for measuring the semantic similarity between sentences or very short texts, based on semantic and word order information. Using corpus statistics and a structured lexical database, this technique calculates the semantic similarity of two sentences. This method fails to disambiguate word sense using surrounding words. Han et al. (2013) developed a new metric pointwise mutual information (PMI), PMI_{max} that augments traditional PMI to improve semantic similarity between two concepts by estimating information about the word's number of senses. PMI_{max} cannot be applied to the field of semantic acquisition from text without its combining with distributional similarity.

Bollegala et al. (2011a) proposed lexical pattern extraction and pattern clustering algorithms to extract the numerous semantic relations that exist between two words. First, various word co-occurrence measures are defined using page counts and then text snippets are used to extract lexical patterns. This technique outperforms the traditional semantic similarity measures by achieving a high correlation and significantly improves the accuracy in a community mining task.

Bollegala et al. (2013) devised a relation extraction method based on latent relational mapping that trains an existing relation extraction system (source relation) to extract new relation types (target relation) with minimum supervision. First, context in which these two entities co-occur are used to extract lexical and syntactic patterns. Next, by constructing a bipartite graph, a classifier is trained by using a small number of labelled instances to identify target relation types. It is shown that this method achieves a statistically significant F-score. But this technique does not handle unrelated entities and multiple semantic relations.

Cilibrasi and Vitanyi (2007) proposed a new similarity measure called Google similarity distance to automatically extract similarity of words and phrases from web using Google page counts. It is shown that this measure results in a significant mean agreement rate. Liu et al. (2010) proposed keyword extraction algorithm using PageRank for ranking of synonyms. First, a weighted co-occurrence network is used to represent the contents of a single document and in this way ranks are assigned for each synonym group using the PageRank algorithm. Lastly, several synonym groups with a high ranking are selected as keywords of the document. It is shown that this algorithm is more adaptable to different types of classes, domains or languages.

Green (1999) designed a lexical chaining method for generating hypertext links both within and between documents based on semantic similarity of words and takes into account the effects of synonymy and polysemy to build hypertext links. First, the lexical chain are analysed, next similarity of paragraphs in these chains are computed. Then, it decides which paragraphs should be linked based on similarities computed and connections examined among them and finally, the hypertext links are built based on these connections. By using this method, the hypertext links are built with significant accuracy.

3 Background and system architecture

3.1 Background

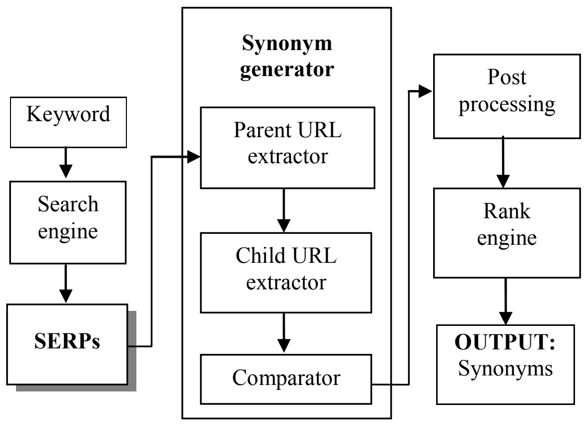
Cheng et al. (2012) proposed an offline, fully automated and data-driven algorithm called identifying normalisation entity synonym that mines queries for instances where a variety of keywords have been used to refer to the same web pages and generates an expanded set of equivalent strings called entity synonyms for each original keyword. This framework consists of three modules: candidate generation, candidate selection, and noise cleaning. This technique effectively enriches the structured data with the help of these entity synonyms. This algorithm is shown to significantly increase the coverage of structured web queries with good precision. However, it should be noted that this technique is applicable only to offline and structured data and not to the dynamic and unstructured WWW. In our approach, we solve this problem as the ADRS algorithm is capable of working in a dynamic, online environment and it is not domain-specific.

3.2 Proposed system architecture

The proposed architecture of our model is illustrated in Figure 1 and its components are:

- *Search engine*: is the web tool that accepts query keywords as inputs, searches the keywords in its database and provides the pages that contain these keywords as SERPs.
- *Candidate synonym generator*: is the main component that generates candidate synonyms for the given keyword. It consists of the subcomponents: *parent URL extractor*, *child URL extractor* and *comparator*. The parent URL extractor extracts the URLs present in the SERPs of the original query keyword. The child URL extractor extracts the URLs present in the SERPs that are obtained by querying the anchor texts of the parent URLs in the search engine. The comparator compares the parent URLs and the inbound anchor texts of the child URLs and if they are equal, these anchor texts are extracted as a set of candidate synonyms for the original search keyword.
- *Post processing*: During post processing stage, repeating synonyms are removed and parts of speech like nouns, verbs, pronouns, prepositions, adverbs and adjectives are allotted to each word in a synonym by applying part of speech (PoS) tags. Only noun and verb synonyms are considered and the rest are ignored.
- *Rank engine*: ranks the candidates with respect to a given keyword and identifies the correct synonym among the extracted candidate synonyms and assigns a higher rank to it. This is achieved with the help of the candidate synonyms' co-occurrence frequencies and page count-based measures. The end result is a ranked list of synonyms for a given keyword arranged in decreasing order of their relevance which is returned to the user.

Figure 1 System architecture



The working of the proposed method for candidate synonym generation is briefly summarised in Figure 2, which consists of the following activities:

1 User inputs a keyword to the search engine to generate SERPs.

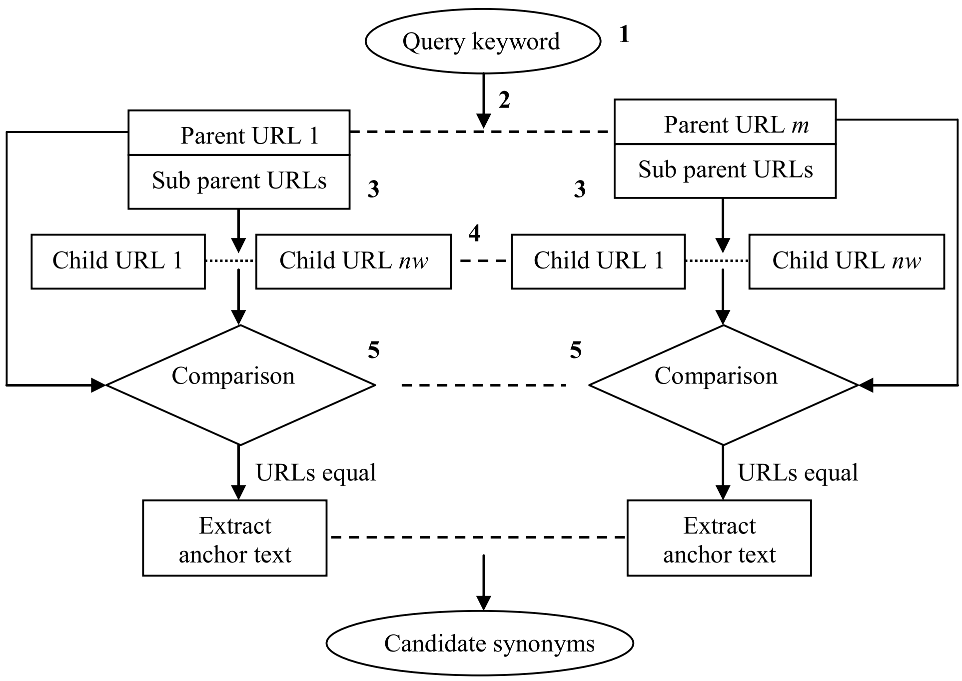
2 *Parent* URL extractor extracts all the URLs contained in SERPs and is collected as a set of *parent* URLs.

3 For each *parent* URL, we retrieve the set of pages that link to *parent* URL from search engine and URLs contained in them are collected as *sub parent* URLs.

4 *Child* URL extractor extracts all the (anchor text, link) pairs by visiting each *sub parent* URL and they are collected as a set of child URLs.

5. Comparator compares each of the *child* URLs with the *parent* URL and when a match occurs, the corresponding inbound anchor text of the *child* URL is the candidate synonym for the keyword and stored in the list.

Figure 2 Model flowchart



4 Problem definition and system model

4.1 Problem definition

Given a keyword A, which is a real world entity and web search engine S, our objective is to extract synonyms for a given keyword from WWW using S and rank candidate synonyms based on co-occurrence frequency (CF) and page count-based measures.

4.2 Assumptions

It is assumed that the user is online and only noun and verb synonyms are considered. For parent URL extraction, we take only the first five SERPs into consideration as it is assumed that only those pages contain relevant information.

4.3 Basic definitions

- *Inbound anchor texts* – refers to a set of anchor texts that are pointing to the same URLs that are relevant to the search keyword.
- *Candidate synonyms* – are a set of inbound anchor texts for a search keyword, which have the same URLs linked to them as that of the search keyword.
- *Parent URLs* – are the URLs present in the SERPs of the original query keyword.
- *Sub parent URLs* – For each *parent* URL, we retrieve the set of pages that link to *parent* URL from search engine and URLs contained in them are collected as sub *parent* URLs.
- *Child URLs* – are the URLs present in the pages corresponding to the *sub parent* URLs.

4.4 System model

4.4.1 Generation of candidate synonyms

First keyword A is sent as a query to the search engine S and it returns the SERPs. The URLs of all SERPs are collected to form a set of parent URLs PU, i.e.,

$$PU = \{u_i \mid u_i \text{ is a } URL \in SERP \text{ and } \forall_i,\ 1 = i = m\}$$

where m is the total number of URLs present in all SERPs for query A. Next, for each *parent* URL $u_i \in PU$, we retrieve the set of pages that are linked to u_i using the search engine or in other words send u_i as a query to the search engine to generate the SERPs corresponding to u_i and the subsequent URLs contained in them are collected to form a set of *sub parent URLs SPU*, i.e.,

$$SPU = \{su_{ik} \mid su_{ik} \text{ is a } URL \in SERP \text{ of } u_i \text{, and } \forall_k\ 1 \le k \le n\}.$$

Where n is the number of URLs in SERPs of u_i. Then, for all sub *parent* URLs $su_k \in SPU$ is visited and all the (anchor text, link) pairs that are contained in them are collected to form a set of *child* URLs CU_i, i.e.,

$$CU_i = \left\{\left(ku_j, ka_j\right) \middle| \forall k\ 1 \le k \le n \text{ and } \forall j\ 1 \le j \le w\right\}.$$

where w is the number of (anchor text, link) pairs retrieved for all URLs in SPU.

Finally, each of these *child* URLs $ku_j \in CU_i$ is compared with its corresponding *parent* URL u_i and when $ku_j = u_j$, then the corresponding anchor text ka_j is the candidate synonym for the keyword A to form a set of candidate synonyms CS_i from URL u_i. This is represented by,

$$CS_i = \left\{ka_j \mid ku_j = u_j, \forall \left(ku_j, ka_j\right) \in CU \text{ and } \forall j, 1 \le j \le w\right\}.$$

Similarly, for the remaining URLs in *PU*, the candidate synonyms can be generated. The candidate synonyms *CS* for the given keyword *A* is the Union of candidate synonyms CS_1, CS_2, …, CS_m, i.e.,

$$CS = CS_1 \cup CS_2 \cup \ldots \cup CS_m$$

Once a set of candidate synonyms are extracted, a PoS like noun, verb, pronoun, preposition, adverb, adjective, etc., is assigned to each word in a synonym by applying PoS tagger and only noun and verb synonyms are considered.

Example 1: A keyword *A = 'Fireblade'*, which is a sports motorcycle manufactured by Honda, is entered as a query in the Google search engine to obtain the SERPs. The URLs contained in the SERPs are collected as a set of *parent* URLs, i.e., *PU* = {http://en.wikipedia.org/wiki/Honda_Fireblade, http://world.honda.com/CBR1000RR/, …}.

For the sake of notation, we shall refer to the URLs in *PU* as u_1, u_2, … that are separately queried in the search engine to generate the SERPs and the new URLs contained in them are collected as *sub parent URLs*, (SPU_1 from u_1, SPU_2 from u_2, …)

$$SPU_1 = \{http://en.wikipedia.org/wiki/2008\_Isle\_of\_Man\_TT,$$
$$http://en.wikipedia.org/wiki/2003\_Superbike\_World\_Championship\_season,$$
$$......\},$$

$$SPU_2 = \{http://world.honda.com/siteindex/, ...\}.$$

By visiting all *sub parent* URLs that belongs to the set SPU_1, SPU_2, … the (*anchor text, hyperlink*) pairs that are contained in them are collected as a set of *child* URLs CU_1, CU_2, …, respectively, i.e.,

$$CU_1 = \left\{(http://en.wikipedia.org/wiki/Honda\_Fireblade, Honda\ Fireblade),\right.$$
$$(http://en.wikipedia.org/wiki/Honda\_Fireblade, Honda\ CBR954RR),$$
$$\left.(http://en.wikipedia.org/wiki/Honda\_RC51, HondaVTR1000SPW)\right\}.$$

$$CU_2 = \left\{(http://world.honda.com/CBR1000RR/, CBR1000RR)\right\}.$$

Ultimately u_1 is compared with the URL part of the elements of CU_1 and u_2 is compared with the URL part of the elements of CU_2 and so on. Since u_1 is the same as the URL part of the first element of CU_1, the corresponding anchor text part 'Honda Fireblade' is one of the candidate synonyms. Similarly, this process is repeated for the entire set of *parent* URLs and that synonyms are stored as a list, i.e.,

$$CS_1 = \{Honda\ Fireblade, Honda\ CBR954RR\}$$

$$CS_2 = \{CBR1000RR\}.$$

This process is repeated for the entire set of *parent* URLs. Ultimately, *CS* contains *Honda Fireblade*, *Honda CBR954RR* and *CBR1000RR* as candidate synonyms for the keyword 'Fireblade', i.e., *CS* = {*Honda Fireblade, Honda CBR954RR and CBR1000RR*}. ■

4.4.2 Ranking of candidate synonyms

For a given keyword *A*, many candidate synonyms are generated and the most relevant among these candidates must be ranked in the first position. This ranking can be achieved by finding the association between a keyword and the candidate synonyms. To compute this association, two techniques: CF and page counts-based measures, can be employed to rank the candidate synonyms.

- CF

  CF between a unique keyword *A* and its candidate synonym $c_i \in$ CS is the number of distinct URLs that are linked to *A* and c_i. The higher the CF, the more relevant the candidate synonym and so the CF is computed between A and all its candidate synonyms and ranked based on the decreasing order of the CFs. Hence the candidate with the highest CF value will be ranked in the first position and so on.

Example 2: CFs between the keyword 'Fireblade' and its candidate synonyms generated in Example 1 are: (Fireblade, Honda Fireblade) is 3, (Fireblade, Honda CBR954RR) is 2 and (Fireblade, CBR1000RR} is also 1. As (Fireblade, Honda Fireblade) has the highest CF value, Honda Fireblade is the most relevant synonym for the keyword 'Fireblade' and ranked in the first position and the other two synonyms are ranked in the second and third positions respectively. ■

- Page count-based measures

  The page count of a query keyword *A* is an estimate of the number of pages that contains the query keyword. The candidate synonyms are ranked using similarity measures that in turn use page counts of *A* only, c_i only and both *A* and c_i. Let N_A be the page counts for keyword *A only*, Nc_i be the page counts for a candidate synonym c_i only and N_{Aci} be the page counts for both keyword *A* and a candidate synonym c_i.

  A similarity measure is a function which computes the degree of similarity and represents the similarity between two words. We use nine popular *relevance* measures: WebJaccard, Cosine, WebDice, WebOverlap, Precision, Recall, WebPMI (PMI) and normalised Google distance (NGD), to accurately measure the relevance between *A* and c_i and to rank the candidate synonyms using page counts. These similarity measures are described using N_A, Nc_i and N_{Aci} as follows:

  1 *WebJaccard coefficient* is the measure used for comparing the similarity and diversity of sample sets and is defines as the size of the intersection divided by the size of the union of the sample sets. WebJaccard coefficient between keyword *A* and a candidate synonym $c_i \in CS$ is defined as,

  $$WebJaccard\left(A, c_i\right) = \frac{N_{Ac_i}}{N_A + N_{c_i} - N_{Ac_i}}$$

  2 *Cosine similarity* is a measure of similarity between two vectors that measures the cosine of the angle between them. Cosine similarity between keyword *A* and a candidate synonym $c_i \in CS$ is defined as,

  $$\cos\left(A, c_i\right) = \frac{N_{Ac_i}}{\sqrt{N_A}\sqrt{N_{c_i}}}$$

3 *WebDice* (A, c_i) between keyword A and a candidate synonym $c_i \in CS$ is defined as,

$$WebDice(A, c_i) = \frac{2N_{Ac_i}}{N_A + N_{c_i}}$$

4 *WebOverlap* (A, c_i) between keyword A and a candidate synonym $c_i \in CS$ is defined as,

$$Overlap(A, c_i) = \frac{N_{Ac_i}}{\min(N_A, N_{c_i})}$$

5 *F-score* can also be used to rank the candidate synonyms. F-score of a synonym c_i is computed as the harmonic mean between the precision and recall of the synonym. First, for a synonym c_i and a keyword A, we compute its precision and recall as follows:

a Precision is the ratio between the number of relevant documents that are returned and the total number of returned documents. *Precision* (A, c_i) between 'A and c_i' is defined as,

$$Precision(A, c_i) = \frac{N_{Ac_i}}{N_A}$$

b Recall is the ratio between the number of relevant documents that are returned and the total number of relevant documents. *Recall* (A, c_i) between 'A and c_i' is defined as,

$$Recall(A, c_i) = \frac{N_{Ac_i}}{N_{c_i}}$$

Then, its F-score is computed as,

$$F - score = \frac{2 \times \text{Precision}(A, c_i) \times \text{recall}(A, c_i)}{\text{Precision}(A, c_i) + \text{recall}(A, c_i)}$$

If L is the number of documents indexed by the search engine *WebPMI* between keyword A and a candidate synonym $c_i \in CS$ is defined as,

$$WesPMI(A, c_i) = \log_2\left(\frac{LN_{Ac_i}}{N_A N_{c_i}}\right)$$

NGD is a semantic similarity measure derived using page counts retrieved from Google search engine. The lesser the NGD between two words, higher the similarity and vice versa. If L is the number of documents indexed by the search engine, then *NGD* (A, c_i) between keyword A and a candidate synonym $c_i \in CS$ is defined as,

$$NGD(A, c_i) = \frac{\max(\log N_A, \log N_{c_i}) - \log N_{Ac_i}}{\log L - \min(\log N_A, \log N_{c_i})}$$

Table 1 ADRS algorithm

Input: Keyword A
Output: List of Synonyms for A, ranked in the order of relevance.
//Generation of Candidate Synonyms
SERPs = SearchEngine(A)
PU = ExtractAllURLs(SERPs)
SPU = Ø
CS = Ø
For each URL $u_j \in$ PU **do**
    SPU_i = get all pages that contains URL u_j
    SPU = $\cup_i$ SPU_i
    CU_i = Ø
    For each URL $su_k \in$ SPU **do**
        P = Visit the Web page corresponding to URL su_k
        CU_{ik} = collect all anchortext and URL pair (ku_j, ka_j) from P;
        $CU_{ik} = \cup_{ik} CU_{ik}$
    End for
    For each $ku_i \in$ CU **do**
        If ($u_i == ku_j$) **then**
            $CS_i \cup_i ka_i$
        End if
    End for
    CS = $\cup_i$ CS_i
End for
//Ranking Candidate Synonyms
For each URL $c_i \in$CS **do**
    Calculate co-occurrence frequency A and c_i
    Compute the page counts for A (N_A)
    Compute the page counts for A and c_i (N_{Aci})
    Compute the page counts for c_i (N_{ci})
    Rank candidate synonym $c_i \in$ CS
    WebJaccard(A, c_i) = $N_{Aci}/(N_A + N_{ci} - N_{Aci})$
    Cos(A, c_i) = N_{Aci}/Sqrt(N_A)Sqrt(N_{ci})
    WebDice(A, c_i) = (2 N_{Aci})/($N_{ci} + N_A$)
    WebOverlap(A, c_i) = N_{Aci}/min(N_A, N_{ci})
    Precision(A, c_i) = N_{Aci}/N_A
    Recall(A, c_i) = N_{Aci}/N_{ci}
        F-Score = (2 × Precision(A, c_i × Recall(A, c_i))/(Precision(A, c_i) + Recall(A, c_i))
        WebPMI(A, c_i) = $\log_2$(L × $N_{Aci}/N_A \times N_{ci}$)
        NGD(A, c_i) = (max{log N_A, log N_{ci}} – log N_{Aci})/(log L – min{log N_A, log N_{ci}})
End for

4.5 ADRS algorithm

The ADRS algorithm is shown in Table 1 which extracts synonyms from the web and is based on the described model. In this algorithm, the search engine first uses the keyword to generate the SERPs and the URLs contained in them are collected as a set of *parent* URLs. A hash map that stores the URLs of the pages that contain links to these *parent* URLs is maintained as SPU. Next, each page of SPU is visited and all the (anchor text, link) pairs that are contained in it are collected as a set of *child* URLs for this *parent*. Finally, each of these *child* URLs are compared with the *parent* URL and when a match occurs, the inbound anchor text of the *child* URL is extracted and stored in the hash map with the *parent* URL. This process is repeated for the entire set of *parent* URLs and ultimately the hash map contains candidates for synonyms for the keyword. The next stage of the algorithm is to rank the candidate synonyms based on the CF and page count-based measures.

5 Experimental results

The ADRS algorithm has been implemented using Java language using the Netbeans 7.1 IDE and Google search engine in a Pentium Dual Core processor environment, with 2 GB of RAM and 100 GB HDD. Experiments were conducted to derive the candidate synonyms for some popular search keywords such as titles of movies, camera models, motorcycle models and place names. For parent URL extraction, only the first five SERPs have been considered and for the sub parent and child URLs, all the SERPs have been taken into consideration.

Table 2 lists the query keywords used and their synonyms along with their CFs. The synonyms with the highest CFs are listed first and are hence the most relevant synonyms associated with those keywords. Some synonyms have the same CF value which implies that their relevance is the same and can be listed in any order.

Tables 3 and 4 specifically show the synonyms for the keywords, ‘Bengalooru’ and ‘Armour of God’ respectively. Here, the page count-based measures have been used to evaluate the similarity between the keywords and their synonyms to compare the ranking of these synonyms under these measures. From Table 3, it can be observed that the synonym ‘Bangalore’ has the highest similarity score compared to the other synonyms with respect to all the measures mentioned and hence it is the most relevant synonym for the keyword ‘Bengalooru’. ‘Bengaluru’ has the second highest similarity score and ‘IT Capital of India’ the least. Similarly, from Table 4, the synonym ‘Armour of God (film)’ has the highest similarity score for the keyword ‘Armour of God’.

To compare the similarity measures, we have used mean reciprocal rank (MRR) (Bollegala et al., 2008). If r_i is the rank of the correct synonym c_i and n is the total number of candidate synonyms, then MRR is defined as,

$$MRR = \frac{1}{n}\sum_{i=1}^{n}\frac{1}{r_i}$$

Table 5 lists the MRR in descending order for the keywords *Bengalooru and Armour of God* for different measures. It can be observed that correct synonyms are obtained by using the proposed algorithm with WebJaccard.

Table 2 Common keywords and their synonyms

Keywords	*Candidate synonyms with CFs*
Danny the Dog	Unleashed (4)
	Unleashed (aka Danny the Dog) (2)
	Unleashed/Danny the Dog (1)
	Danny the Dog (2005) (1)
Fireblade	Honda Fireblade (3)
	CBR1000RR (1)
	Honda CBR1000RR (1)
Live Free or Die Hard	Die Hard 4 (3)
	Die Hard 4.0 (2)
	Live Free or Die Hard (2007) (2)
	Die Hard IV (1)
Armour of God	Operation Condor 2: The Armour of the Gods (1)
	Long Xiong Hu Di (1)
	Armour of God (film) (1)
Welcome to the jungle	The Rundown (1)
	The Rundown (Welcome to the Jungle) (1)
	Guns N' Roses – Welcome to the Jungle (1)
Canon EOS 400D	Canon EOS Kiss Digital X (1)
	EOS Rebel XTi (1)
	400D Digital Rebel XTi Kiss Digital X (1)
Bengalooru	Bangalore (2)
	Bengaluru (1)
	Garden City (1)
	IT Capital of India (1)
US	USA (2)
	United States of America (1)
Republic of India	India (2)
	Bharat (1)
Mac OS X	Leopard (2)

Table 3 Ranked synonyms for 'Bengalooru'

Synonym	*WebJaccard*	*Cosine*	*WebDice*	*WebOverlap*	*F-score*	*NGD*
Bangalore	0.1275	0.396	0.2261	1.263	0.2262	0.7098
Bengaluru	0.0141	0.1188	0.0279	1	0.0279	0.6598
Garden city	0.0029	0.0535	0.0057	1.0094	0.0058	0.4541
IT capital of India	0.0073	0.0145	0.0144	0.0154	0.0145	0.1387

Table 4 Ranked synonyms for 'Armour of God'

Synonym	*WebJaccard*	*Cosine*	*WebDice*	*WebOverlap*	*F-score*	*NGD*
Armour of God (film)	0.0088	0.1188	4.0262	6.8824	1.1327	0.6462
Operation Condor 2: The Armour of the Gods	0.0098	0.0535	0.0115	0.8806	0.0154	0.6289
Long Xiong Hu Di	0.0123	0.0145	0.0476	0.1903	0.0244	0.5919

Table 5 MRR scores for page count-based measures

Measure	*MRR (Bengalooru)*	*MRR (Armour of God)*
WebJaccard	70.07238	37.12225
WebDice	35.64352	13.52666
F-score	35.20531	13.35019
Cosine	12.32498	12.00933
WebOverlap	8.46469	0.816969
NGD	1.542053	0.603383

6 Conclusions

We have proposed a scalable algorithm called ADRS that generates a ranked list of candidate synonyms for a search keyword and that works even in the online, dynamic, domain-independent web. Our method returns a ranked and accurate list of URLs for a given keyword. ADRS effectively tackles the *polysemy* problem by providing the user with valuable and correct synonyms for the given keyword. The experimental results have shown that correct synonyms were obtained using the proposed algorithm with WebJaccard. As a future enhancement to ADRS, snippets in the SERPs can also be taken into consideration for generating synonyms.

References

Ageishi, R. and Miura, T. (2010) 'Automatic extraction of synonyms based on statistical machine translation', *22nd International Conference on Tools with Artificial Intelligence*, pp.313–317.

Balog, K., Azzopardi, L. and Azzopardi, L. (2008) 'Personal name resolution of web people search', *NLPIX2008*, Beijing, China, 22 April.

Bollegala, D., Matsuo, Y. and Ishizuka, M. (2008) 'A co-occurrence graph-based approach for personal name alias extraction from anchor texts', *International Joint Conference on Natural Language Processing*, pp.865–870.

Bollegala, D., Matsuo, Y. and Ishizuka, M. (2011a) 'A web search engine-based approach to measure semantic similarity between words', *IEEE Transactions on Knowledge and Data Engineering*, July, Vol. 23, No. 7, pp.977–990.

Bollegala, D., Matsuo, Y. and Ishizuka, M. (2011b) 'Automatic discovery of personal name aliases from the web', *IEEE Transactions on Knowledge and Data Engineering*, June, Vol. 23, No. 6, pp.831–844.

Bollegala, D., Matsuo, Y. and Ishizuka, M. (2013) 'Minimally supervised novel relation extraction using a latent relational mapping', *IEEE Transactions on Knowledge and Data Engineering*, February, Vol. 25, No. 2, pp.419–432.

Cheng, T., Lauw, H.W. and Paparizos, S. (2012) 'Entity synonyms for structured web search', *IEEE Transactions on Knowledge and Data Engineering*, October, Vol. 24, No. 10, pp.1862–1873.

Cilibrasi, R.L. and Vitanyi, P.M.B. (2007) 'The Google similarity distance', *IEEE Transactions on Knowledge and Data Engineering*, March, Vol. 19, No. 3, pp.370–383.

Green, S.J. (1999) 'Building hypertext links by computing semantic similarity', *IEEE Transactions on Knowledge and Data Engineering*, September/October, Vol. 11, No. 5, pp.713–730.

Han, L., Finin, T., McNamee, P., Joshi, A. and Yesha, Y. (2013) 'Improving word similarity by augmenting PMI with estimates of word polysemy', *IEEE Transactions on Knowledge and Data Engineering*, June, Vol. 25, No. 6, pp.1307–1322.

Harada, M., Sato, S-y. and Kazama, K. (2004) 'Finding authoritative people from the web', *Proceedings of the Joint ACM/IEEE Conference on Digital Libraries (JCDL'04)*, pp.306–313.

Iosif, E. and Potamianos, A. (2010) 'Unsupervised semantic similarity computation between terms using web documents', *IEEE Transactions on Knowledge and Data Engineering*, November, Vol. 22, No. 11, pp.1637–1647.

Jiang, L., Wang, J., An, J., Wang, S., Zhan, J. and Li, L. (2009) 'GRAPH: a graph-based framework for disambiguating people appearances in web search', *Ninth IEEE International Conference on Data Mining*, pp.199–208.

Kalashnikov, D.V., Chen, Z., Mehrotra, S. and Nuray-Turan, R. (2008) 'Web people search via connection analysis', *IEEE Transactions on Knowledge and Data Engineering*, November, Vol. 20, No. 11, pp.1–16.

Kawai, Y., Yoshikawa, T., Furuhashi, T., Hiraoy, E., Kunoy, A. and Gotohy, T. (2012) 'A study on extraction method of synonyms in specification documents', *13th ACIS International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing*, pp.321–324.

Lefever, E., Fayruzov, T., Hoste, V. and De Cock, M. (2009) 'Fuzzy ants clustering for web people search', *2nd Web People Search Evaluation Workshop (WePS 2009); 18th WWW Conference*, Madrid, Spain, April.

Li, Y., Bandar, Z.A. and McLean, D. (2003) 'An approach for measuring semantic similarity between words using multiple information sources', *IEEE Transactions on Knowledge and Data Engineering*, July/August, Vol. 15, No. 4, pp.871–882.

Li, Y., McLean, D., Bandar, Z.A., O'Shea, J.D. and Crockett, K. (2006) 'Sentence similarity based on semantic nets and corpus statistics', *IEEE Transactions on Knowledge and Data Engineering*, August, Vol. 18, No. 8, pp.1138–1150.

Liu, Z., Liu, J., Yao, W. and Wang, C. (2010) 'Keyword extraction using PageRank on synonym networks', *ICEEE*, Henan, 7–9 November, pp.1–4.

Niemi, J., Lindén, K. and Hyvarinen, M. (2012) 'using a bilingual resource to add synonyms to a Wordnet: FinnWordNet and Wikipedia as an example', *6th International Global Wordnet Conference*, Matsue Japan, January, pp.227–231.

Shen, D., Walker, T., Zheng, Z., Yang, Q. and Li, Y. (2008) 'Personal name classification in web queries', *WSDM'08*, Palo Alto, California, USA, 11–12 February, pp.149–158.

Shen, Q. and Boongoen, T. (2012) 'Fuzzy orders-of-magnitude-based link analysis for qualitative alias detection', *IEEE Transactions on Knowledge and Data Engineering*, April, Vol. 24, No. 4, pp.649–662.

Simanovsky, A. and Ulanov, A. (2011) 'Mining text patterns for synonyms extraction', *22nd International Workshop on Database and Expert Systems Applications*, pp.473–477.

Smirnova, E., Avrachenkov, K. and Trousse, B. (2010) 'Using web graph structure for person name disambiguation', in *Third Web People Search Evaluation Forum (WePS-3), CLEF Conference*.

Takeuchi, K. (2008) 'Extraction of verb synonyms using co-clustering approach', *Second International Symposium on Universal Communication*, pp.173–178.

Van der Plas, L. and Tiedemann, J. (2006) 'Finding synonyms using automatic word alignment and measures of distributional similarity', *Annual Meeting of the Association of Computational Linguistics*, pp.866–873.